One of the most frequently asked questions on the Internet site for the 80C32-BASIC control computer is, ‘Can it burn EPROMs?’ In fact, it can, and this article explains how.

Design by H. J. Böhling

EEPROMs and the BASIC control computer

FAQs for the BASIC computer

The 8052AH-BASIC V1.1 processor uses port 0 for programming EPROMs. Since this port is also used for the external data and address bus, using it to burn EPROMs is only possible because the BASIC interpreter is stored in the internal memory of the 8052. It thus does not need the external data/address bus while it is programming an EPROM. Unfortunately, an 80C32 processor, with MCS-51 BASIC located externally, cannot burn EPROMS or EEPROMS! If you need to do this, you will have to have either the original 8052AH processor or an 87C52 processor (with 8 kB of internal EPROM) that has been programmed with the BASIC interpreter. Since the first of these controllers is no longer manufactured by Intel, it is likely to be difficult to obtain. The second controller, manufactured by Philips, is available and even has a lower current consumption, but it must be programmed with MCS-51 BASIC before it can be used. If you are not able to do this yourself, you can obtain a preprogrammed controller from Elektor Readers Services.

In addition, you will need some hardware that switches the programming voltage to the EPROM and controls the low-address register during the programming process. Both of these functions are found on the *Elektor Electronics* 80C32 BASIC Computer, for example (see the Elektor book *Short Course 8051/8032 Microcontrollers and Assembler*, which is available along with the circuit board 910042-1 through our Readers Services).

On the other hand, if you can do without burning EPROMs and can limit yourself to 8 kB EEPROMS, it's also possible to work with the 80C32-BASIC control computer. This still requires a processor with an internal BASIC interpreter, plus a number of modifications to the hardware and the circuit board (980002-1), as follows:

- ✓ Remove the 80C32 CPU (IC1) from its socket.
- ✓ On the top side of the circuit board, cut the track at pin 31 of the CPU socket (IC1).
- ✓ Connect a 47 kΩ resistor (R2) between pin 31 of the CPU socket and pin 40 of the CPU socket (+5 V).
- ✓ On the bottom side of the circuit board, cut the track that connects pin 30 of the CPU (ALE/P) to pin 11 of the address register (IC2). Solder a 1N4148 diode (D1) across the break, with the cathode on the CPU side and the anode on the IC2 side.
- ✓ Connect a 47 kΩ resistor (R3)

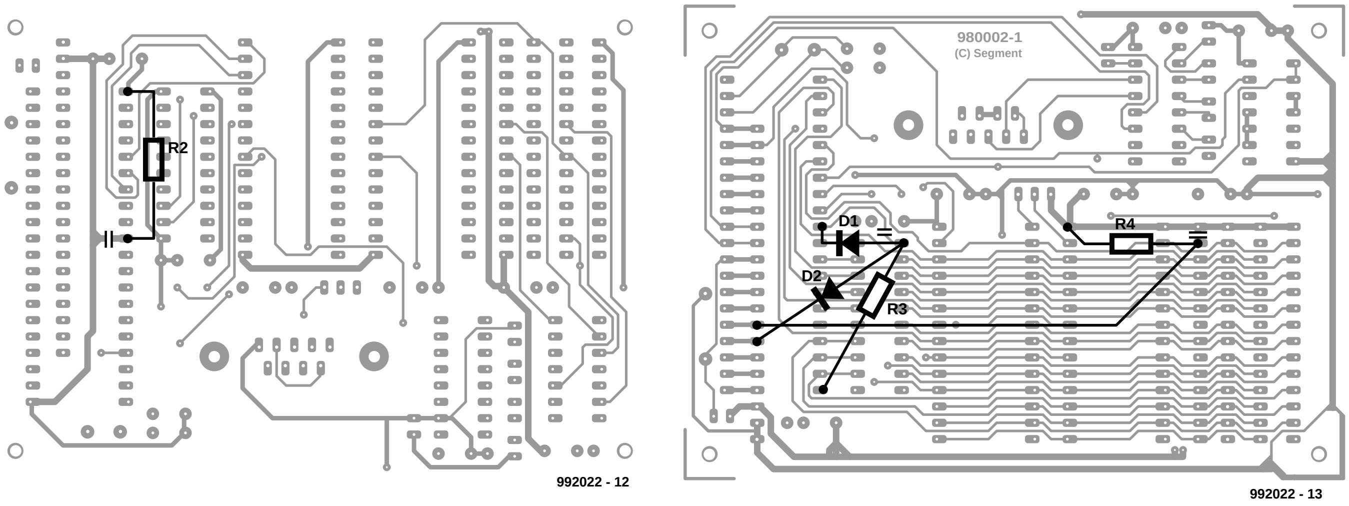

Figure 1. Modifications to the top and bottom sides of the printed circuit board.

between pin 11 of IC2 and pin 40 of the CPU socket (+5 V).

- ✓ Connect a 1N4148 diode (D2) between pin 11 of IC2 and pin 4 of the CPU, with the cathode at the CPU end and the anode at the IC2 end.
- ✓ On the bottom side of the circuit board, cut the track that connects pins 27 and 28 of IC5.
- ✓ Connect a 47 kΩ resistor (R4) between pin 27 of IC5 and pin 28 of IC5 (+5 V).
- ✓ Connect a length of insulated wire between pin 27 of IC5 and pin 5 of the CPU (P1.4).
- ✓ Install a CPU IC with internal 8052AH BASIC in the IC1 socket.
- ✓ Insert a 28C64 EEPROM in the IC5 socket.
- ✓ The BASIC EPROM (IC4) is not needed and may be removed.

How it works

Pin 31 of the CPU is pulled high via resistor R2. This activates the internal ROM of an 8052-series processor. This means that external program memory can no longer be accessed in the address range of 0 to $1FFF_H$. The memory region from 2000_H to $7FFF_H$ can of course still be used for BASIC extensions and machine-language programs. The signal ALE/P is ANDed with the signal ALEDIS (P1.3 of the CPU) via D1, D2 and R3. The resulting signal drives pin 11 of the address register IC2. The signal ~~PGM~~ (P1.4 of the CPU), with its pull-up resistor R4, drives the Write input of the EEPROM. Unfortunately, not every 28C64 EEPROM can be used with this arrangement — in particular, ST Microelectronics EEPROMs with the type number M28C64C-15P1 (manufacturer code ST) cannot be written!

Question time

How do I develop a program and burn an autostart EEPROM?

The program is first developed in the RAM of the control computer. This can be done using the terminal emulator program Terminal MCS-51, which can be found at various places, including the 80C32-BASIC control computer home page at

http://www.germany/net/teilnehmer/101.107378/index.htm

and on the Elektor CD-ROM **986007-1**, the Reuss Mikrocontroller V1.0 CD-ROM and the Pegasus Vol. 8 CD-ROM.

Terminal MCS-51 has a many advantages compared to conventional terminal emulators, such as HyperTerminal. For example, it includes a convenient line editor and the ability to store programs in the PC or load them into the control computer by simply pressing a button.

An autostart program should service the watchdog timer via the timer interrupt. The program CLOCK.LIS can be used as a starting point. You can simply insert your own program code between lines 40 and 59999. After the program is complete, it can be downloaded from the PC to a BASIC control computer that has been modified as described above. The MCS-51 BASIC command PROG causes the downloaded program to be burned as the first program in an empty EEPROM, and the command PROG2 converts it into an autostart program.

If the execution of the autostart program is for some reason suspended and the watchdog function has been activated by removing jumper JP1 on the peripheral circuit board, a reset will be generated after around 10 seconds. This causes the program to restart from the beginning. Don't forget that the command PROG2 also stores the terminal baud rate in the EEPROM!

How do I erase an EEPROM?

A programmed EEPROM can be erased using the small MCS-51 BASIC program ERASE.LIS. This program writes the value $0FF_H$ (255) to every location of the EEPROM. Completely erasing an EEPROM takes about two minutes.

(992022-1)

```
10     REM ***********************************
20     REM * ERASE for erasing EEPROMs       *
30     REM *    (C) H.-J. Boehling 21.08.98  *
40     REM ***********************************
50     PRINT ”Preparing to erase EEPROMs.”
60    MTOP=5FFFH :  REM save RAM memory from 6000H (upper 8 Kbyte)
70     REM ===== fill RAM with FFH ==========
80     FOR I=6000H TO 7FFFH : XBY(I)=0FFH :  NEXT I
90     REM ===== load registers ===============
100    REM source = 6000H to 7FFFH (RAM) to
110    REM target   = 8000H to 9FFFH (EEPROM)
120    REM length = 2000H Bytes (8 Kbytes)
130   DBY(19H)=000H :  REM source LSB
140   DBY(1BH)=060H :  REM source HSB
150   DBY(18H)=0FFH :  REM target - 1 LSB
160   DBY(1AH)=07FH :  REM target - 1 HSB
170   DBY(1EH)=000H :  REM length LSB
180   DBY(1FH)=020H :  REM length HSB
190   DBY(26H)=DBY(26H).AND.0F7H :  REM standard programmming
200   DBY(40H)=0DCH :  REM program pulse length 10 ms LSB
210   DBY(41H)=000H :  REM program pulse length 10 ms HSB
220   XBY(128H)=0DCH :  REM program pulse length for PROG LSB
230   XBY(129H)=000H :  REM program pulse length for PROG HSB
240    REM ===== Prompt: erase? ========
250    PRINT ”Do you want to erase EEPROM now? (Y/N) :”,
260   K=GET :  IF K=0 THEN 260
270    PRINT CHR(K)
280    IF K>90 THEN K=K-32 :  REM convert to upper case
290    IF K=ASC(’Y’) THEN 310
300    GOTO 400
310    PRINT ”Erasing EEPROM! Please wait 2 minutes.” :  PGM
320    REM ===== Error checking ========
330    PRINT
340    IF (DBY(30).OR.DBY(31))=0 THEN 380
350   H=DBY(1AH) : L=DBY(18H) : HL=H*256+L
360    PRINT ”EEPROM not empty at address :”, :  PH1. HL
370    GOTO 400
380    PRINT ”No errors occurred during erase operation.”
390    PRINT ”EEPROM now empty!”
400    REM ===== End =======================
410   MTOP=7FFFH :  REM enable RAM
```

COMPONENTS LIST

Resistors:
R2,R3,R4 = 47kΩ

Semiconductors:
D1,D2 = 1N4148
IC1 = 8052AH-BASIC or 87C52 with MCS-51 BASIC interpreter (order code **996510-1**)
IC5 = 28C64A-20/P (Microchip) or HN58064-25 (Hitachi)